ESCUELA INTERNACIONAL POSGRADOS

Máster en Programación avanzada en Python para Big Data, Hacking y Machine Learning

DESARROLLO SEGURO EN PYTHON

LECCIÓN 01

Introducción al desarrollo de software seguro

ÍNDICE

INTRODUCCIÓN

En esta lección veremos que se entiende por desarrollo de software seguro, las propiedades del software seguro y los principios de diseño seguro

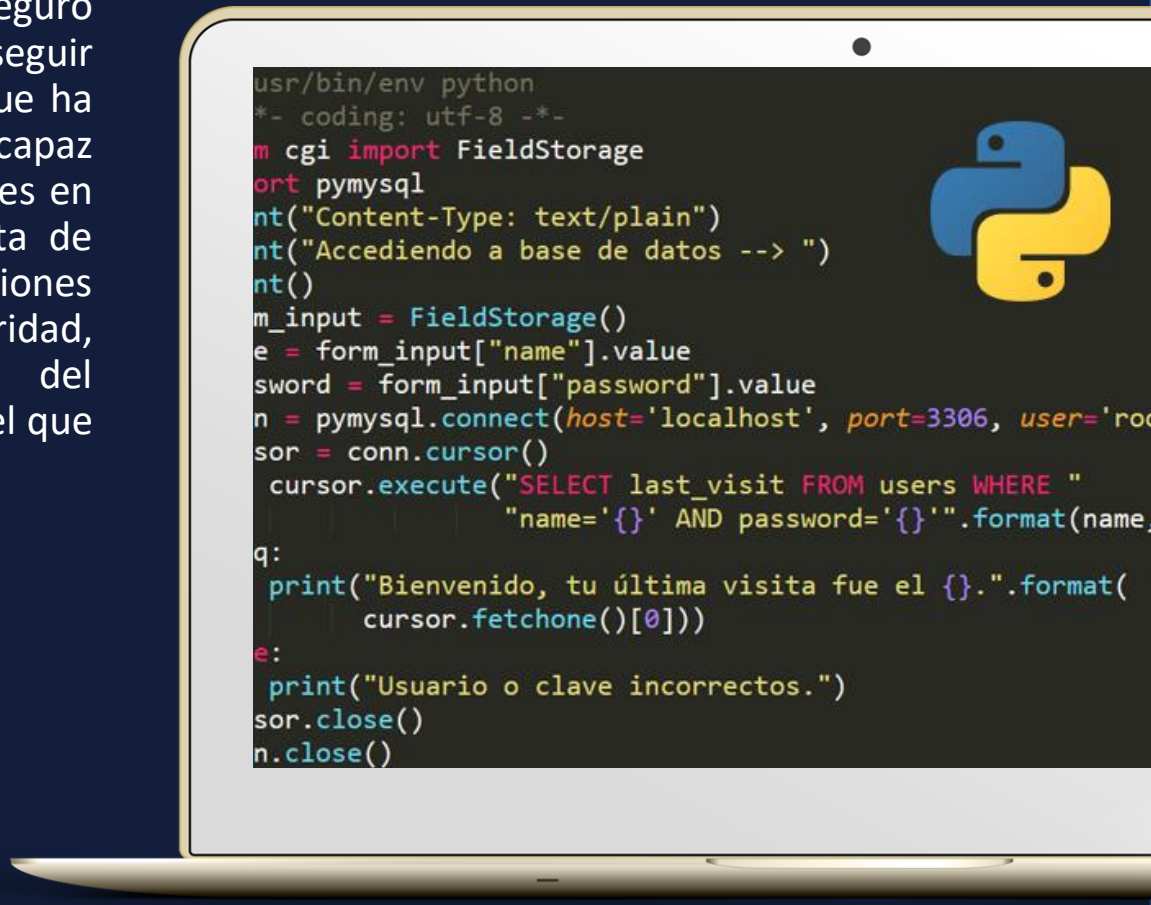
OBJETIVOS

Al finalizar esta lección serás capaz de:

1. Saber en que consiste el desarrollo seguro de software
2. Conocer las propiedades del software seguro
3. Conocer los principios del diseño seguro
4. Incluir el desarrollo seguro en tus proyectos

Definición “Desarrollo Seguro de Software”

Podemos definir un programa seguro como aquel que es capaz de seguir realizando las funciones para las que ha sido diseñado en todo momento, y capaz de evitar que la existencia de errores en él pueda ser utilizada como puerta de entrada para la realización de acciones que pongan en peligro la integridad, confidencialidad o disponibilidad del resto de elementos del sistema en el que se está ejecutando.



Los objetivos de la utilización de prácticas de software seguras son las siguientes:

1 Fallos explotables

Los fallos explotables y otros puntos débiles se eliminan en la mayor medida posible.

2 Desarrolladores

La probabilidad de que los desarrolladores puedan producir errores y vulnerabilidades explotables o puertas traseras en el software se reduce o elimina en gran medida.

3 Resistencia

El software es resistente y tolerante a posibles ataques para apoyar el cumplimiento de la misión de la organización.

El objetivo fundamental de la seguridad informática se basa en preservar los siguientes puntos:

Confidencialidad

El software debe asegurar que cualquiera de sus características, los activos que administra y su contenido son accesibles solo para las entidades autorizadas e inaccesibles para el resto.

Integridad

El software y los activos del sistema solo pueden ser modificados por usuarios autorizados

El objetivo fundamental de la seguridad informática se basa en preservar los siguientes puntos:

Disponibilidad

El software debe estar operativo y ser accesible a sus usuarios autorizados siempre que se requiera. De modo que los usuarios puedan realizar sus tareas de formar correcta y dar cumplimiento a los objetivos de la organización que lo utiliza.

Para las entidades que actúan como usuarios se requieren dos propiedades adicionales

Trazabilidad

Todas las acciones relevantes relacionadas con la seguridad de una entidad que actúa como usuario se deben registrar y trazar con el objetivo de disponer de datos para la realización de auditorías; de esta forma, la trazabilidad debe ser posible tanto durante la ocurrencia de las acciones registradas como a posteriori.

No repudio

Constituye la habilidad de prevenir que una entidad que actúa como usuario desmienta o niegue la responsabilidad de acciones que han sido ejecutadas.

Algunos ejemplos de ataques y las propiedades que vulneran

Inyección de SQL		Confidencialidad
Cross-site scripting (XSS)		Integridad, Disponibilidad
Desbordamiento de búfer que inyecta código con el objetivo de obtener y modificar la información de usuarios.		Confidencialidad, Integridad, Disponibilidad, Trazabilidad y No repudio.

Principios de diseño seguro

Partiendo de las propiedades básicas anteriores veremos una serie de principios orientados al diseño seguro de aplicaciones:

Minimizar el área de la superficie de ataque

Seguridad por defecto

Privilegios mínimos

Validación de datos de entrada

Principios de diseño seguro

Partiendo de las propiedades básicas anteriores veremos una serie de principios orientados al diseño seguro de aplicaciones:

Defensa en profundidad

Control seguro de errores

Separación de funciones

Evitar la seguridad por oscuridad

Minimizar el área de la superficie de ataque

No usar componentes innecesarios

Cada característica que se añade a una aplicación incrementa su complejidad y aumenta también el riesgo de aplicación en conjunto. Una nueva característica implica un nuevo punto de ataque. Uno de los factores clave para reducir el riesgo de una aplicación recae en la reducción de la superficie de ataque.

Pueden eliminarse posibles puntos de ataque si se deshabilitan módulos o componentes innecesarios para la aplicación.

Por ejemplo, si la aplicación no utiliza el almacenamiento en caché de resultados, sería recomendable deshabilitar dicho módulo. De esta manera, si se detecta una vulnerabilidad de seguridad en ese módulo, la aplicación no se verá amenazada.

Seguridad por defecto

Seguridad implícita

La seguridad debe ir implícita en el propio desarrollo. Tanto en la experiencia del cliente como en las prácticas habituales de los desarrolladores.

No usar opciones de configuración de seguridad reducidas para evitar que dichas configuraciones compliquen el desarrollo. Si el propio desarrollo de la aplicación obliga a cambiar opciones de seguridad se debe realizar una auditoría de seguridad.

Privilegios mínimos

Mínima cantidad de privilegios

Según el principio del mínimo privilegio, se recomienda que las cuentas de usuario tengan la mínima cantidad de privilegios necesarios. Asimismo, se aconseja que los procesos se ejecuten únicamente con los privilegios necesarios para completar sus tareas.

De esta manera, se limitan los posibles daños que podrían producirse si se ve comprometido el proceso.

Si un atacante llegase a tomar el control de un proceso en un servidor o comprometiese una cuenta de usuario, los privilegios concedidos determinarán, en gran medida, los tipos de operaciones que podrá llegar a realizar dicho atacante.

Validación de datos de entrada

No confiar en los datos del usuario

Se ha de garantizar que la aplicación sea robusta ante todas las posibles formas de entrada de datos, ya sean proporcionados por el usuario, por la infraestructura, por entidades externas o por bases de datos.

Una premisa fundamental reside en no confiar en los datos que el usuario pueda introducir, ya que este tiene todas las posibilidades de manipularlos. La debilidad de seguridad más común en aplicaciones es la falta de validación apropiada de las entradas del usuario o del entorno. Esta debilidad origina casi todas las principales vulnerabilidades en las aplicaciones

Control seguro de errores

No mostrar información detallada de errores al usuario

Es necesario controlar las respuestas cuándo se produce algún error y no mostrar en ellas información que pudiera ayudar al atacante a descubrir datos acerca de cómo ha sido desarrollada la aplicación o cómo funcionan sus procedimientos.

La información detallada de los errores producidos no deberías ser mostrada al usuario, sino que tendría que ser enviada al fichero de log correspondiente. El acceso a este fichero log también debe tener un acceso seguro. Una simple página de error 403 indicando un acceso denegado puede decir a un escáner de vulnerabilidades que un directorio existe y puede proporcionar a una atacante información que le permita realizar un mapa de la estructura de directorios de la aplicación.

Defensa en profundidad

Definir estrategia

Consiste en definir una estrategia de seguridad estándar en la que se establezcan varios controles de defensa en cada una de las capas y los subsistemas de la aplicación.

Estos puntos de control ayudan a garantizar que solo los usuarios autenticados y autorizados puedan obtener el acceso a la siguiente capa y sus datos.

Separación de funciones

Roles bien diferenciados

Un punto importante consiste en la separación de funciones entre los distintos perfiles de la aplicación.

Además de aplicar privilegios mínimos sobre los distintos roles de los usuarios que tenga nuestra aplicación hemos de tener bien diferenciadas las funciones que puede hacer cada usuario y evitar que un usuario de bajo nivel puede tener acceso a funciones de un administrador o usuario de nivel superior.

Evitar la seguridad por oscuridad

Es un control débil

La seguridad de una aplicación no debería depender del secreto o confidencialidad de su diseño o implementación. Si se intenta ocultar secretos mediante el uso de nombres de variables engañosos o de ubicaciones de archivos no habituales, no se estará mejorando la seguridad.

La seguridad basada en oscuridad es un control débil, especialmente si se trata del único control. Esto no significa que mantener secretos constituya una mala idea; significa que la seguridad de los sistemas clave no debería basarse, exclusivamente, en mantener detalles ocultos.

CONCLUSIONES

1. La seguridad no es una capa ni un proceso más de la aplicación debe ser parte y estar integrada en el desarrollo desde el principio.

2. Es más costoso no aplicar seguridad en desarrollo software que descubrir una vulnerabilidad o brecha de seguridad en producción.

3. La seguridad informática se basa en preservar los siguientes puntos:

   confidencialidad, integridad, disponibilidad, trazabilidad y no repudio.

MUCHAS GRACIAS POR SU ATENCIÓN

eip ESCUELA INTERNACIONAL POSGRADOS

dtinedo@grupomainjobs.com

Diego Tinedo Rodríguez
linkedin.com/in/diego-tinedo

twitter.com/eiposgrados

facebook.com/eiposgrados

instagram.com/eiposgrados